Session 205

I/O Kit: PCI Drivers and Open Firmware

Mark Tozer
Wayne Flansburg
Josh de Cesare
Steve Martin
Jason Seng

Session Audience

- Generic PCI card hardware developers
- Mac OS X PCI driver writers
- Hardware testers

What Is a Generic PCI Device?

- A generic PCI device:
  - Does not belong to a Framework
  - Is unique to your application
  - Needs your application in order to communicate with your device

Session Agenda

- Open Firmware
- Boot process
- Mac OS X IORegistry
- I/O Kit generic PCI driver
- Q&A with IOKit and Kernel Architects

Session URLs

PCI Web Page
Hardware Q&A
http://developer.apple.com/hardware/pci/

Hardware Technical Notes
http://developer.apple.com/technotes/indexes/hw-a.html

PCI Bus Supplement to IEEE 1275
http://playground.sun.com/1275/home.html#OFDbussupps

Darwin: Open Source Projects
http://publicsource.apple.com/

http://developer.apple.com/events/urls.html

Open Firmware and PCI

Wayne Flansburg
WW DTS

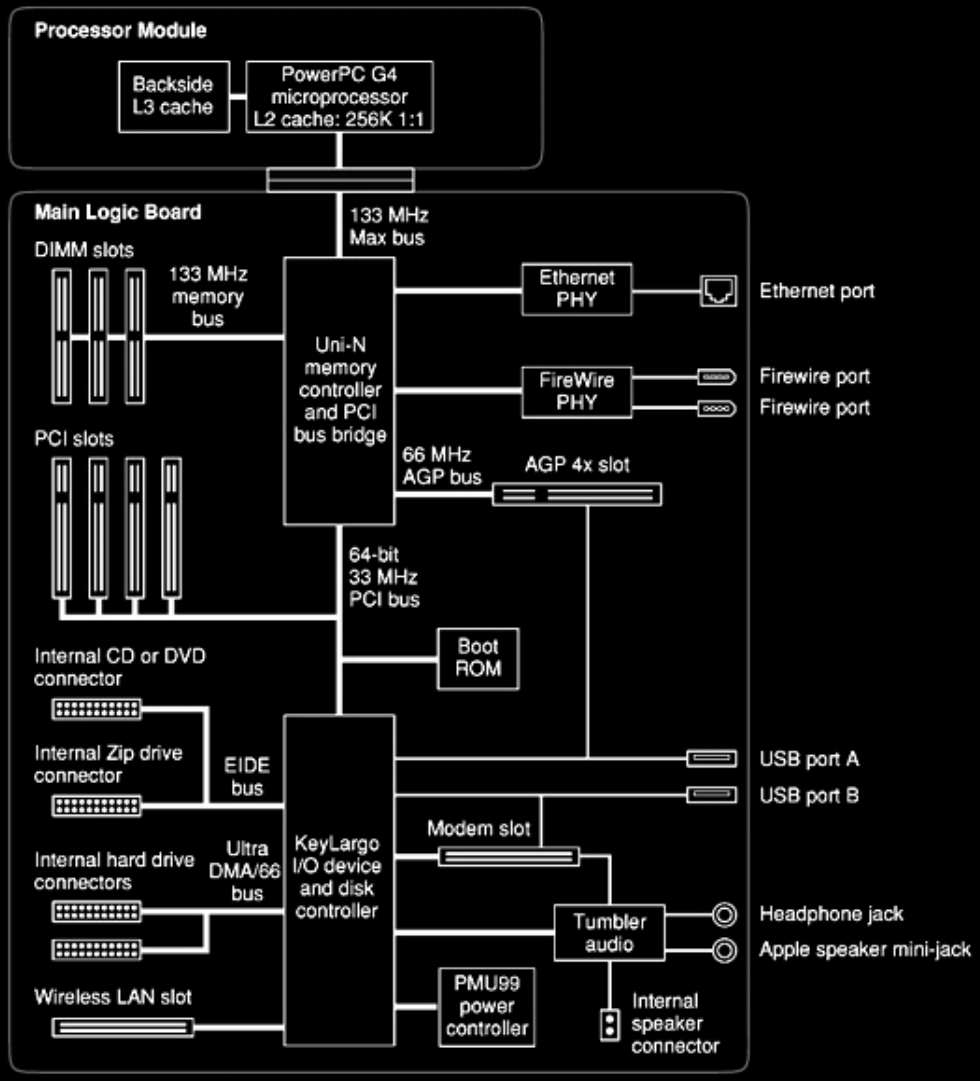
Open Firmware

- The device tree
- Nodes
- Configuration variables
- Configuration space
- Open Firmware security

The Device Tree

- IEEE 1275 required
  - Device tree is a block diagram of the motherboard
  - And much more
- What's in the tree for you?
  - Your node, properties, and words
  - Address to your configuration space, registers, and memory
  - Support nodes
  - Aliases, chosen, options

Block Diagram



Displaying Your Node

- Telnet
- Make your node the active package
  - dev <your node path name>
- Display your words, properties, and children
  - words
  - .properties
  - ls

Reading Your reg property

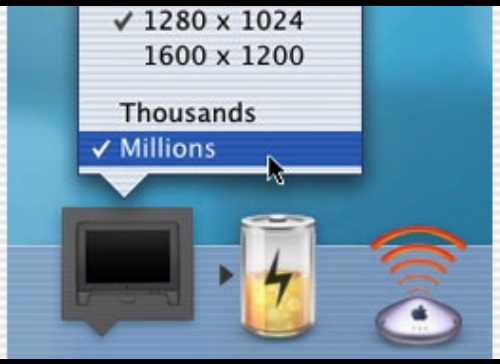
```
        Bit#     33222222 22221111 11111100 00000000
                 10987654 32109876 54321098 76543210

phys.hi cell:    npt000ss bbbbbbbb dddddfff rrrrrrrr
phys.mid cell:   hhhhhhhh hhhhhhhh hhhhhhhh hhhhhhhh
phys.lo cell:    llllllll llllllll llllllll llllllll
```

where:

n	is 0 if the address is relocatable, 1 otherwise
p	is 1 if the addressable region is "prefetchable", 0 otherwise
t	is 1 if the address is aliased (for non-relocatable I/O), below 1 MB (or below 64 KB (for relocatable I/O).
ss	is the space code, denoting the address space
bbbbbbbb	is the 8-bit Bus Number
ddddd	is the 5-bit Device Number
fff	is the 3-bit Function Number
rrrrrrrr	is the 8-bit Register Number
hh...hh	is a 32-bit unsigned number
ll...ll	is a 32-bit unsigned number

Demo

Device Tree Nodes



Steve Martin WW DTS

PCI Configuration Registers

				Hex
Device ID		Vendor ID		00
Status Register		Command Register		04
Class Code			Revision ID	08
BIST	Header Type	Latency Timer	Cache Line Size	0C
Base Address 0				10
CardBus CIS Pointer				28
Subsystem ID		Subsystem Vendor ID		2C
Expansion ROM Base Address				30
Reserved			Capabilities Pointer	34
Reserved				38
Max_Lat	Min_Gnt	Interrupt Pin	Interrupt Line	3C

Required

Optional

Configuration Variables and Support

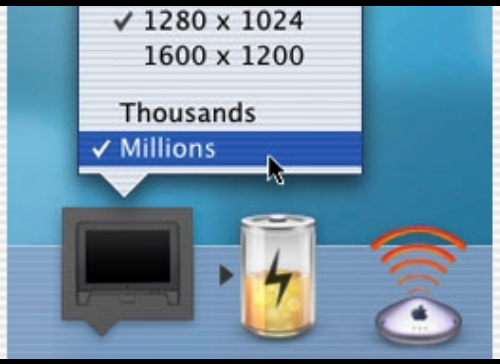
- At the Open Firmware user interface type
  - printenv
- These variables can also be viewed and set from the Terminal application
- Aliases, chosen, and options nodes

Reading Your Configuration Space

- Locate your device
- Display your properties
- Use .cregs to display your registers
- Use phys-hi as the input to .cregs
- Create your own definition for .cregs
  - See Q&A

Demo

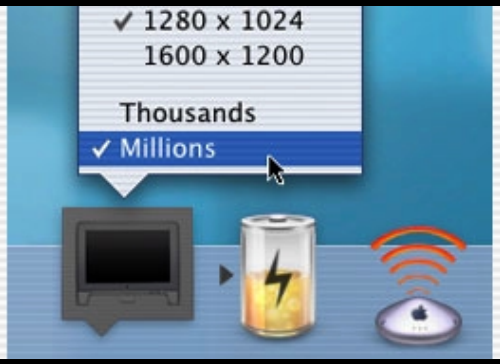
Configuration Space



Open Firmware Security

- Compliant with IEEE 1275
  - security-mode (-- n)
    - Not effected by set-default or set-defaults
  - password (--)
- To reset security change the amount of memory
- Physical lock on Macintosh

Boot Process



Boot Picker Features

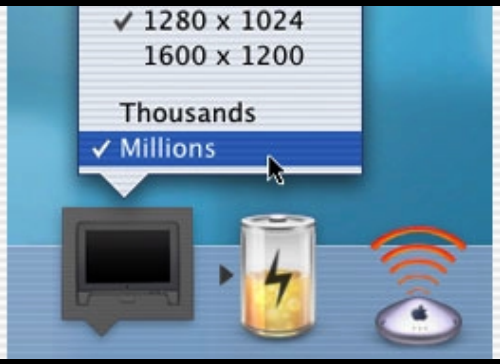
- Boot with option key depressed
- Displays rescan and boot buttons
- Searches device tree for bootable volume
- Displays icon for bootable volumes

Boot Process

- Open Firmware Picker
  - On newer machines
  - May not be on yours until you do a BootROM update
  - Look at the chosen node in the IORegistry to see the boot path

“ /pci@80000000/mac-io@10/ata3@20000/@0:6,\\:tbxi”

Mac OS X IORegistry



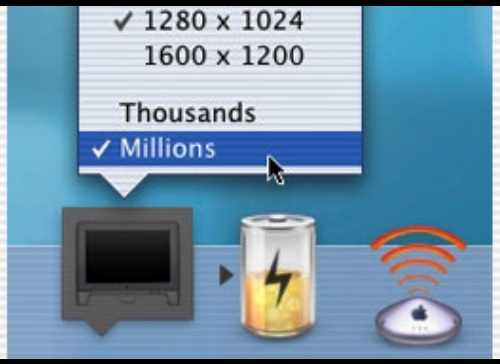
Jason Seng
WW DTS

IORegistry and Its Planes

- IOService
- IOUSB
- IOFireWire
- IOPower
- IOAudio
- IODeviceTree

Demo

IORegistry



View IORegistry Using the Terminal Application

- Determine your provider
- Determine your class
- Read and write configuration variables

Technology Framework Headers

- Inherent from IOService
- Provided by
  - IOPCIDevice
  - IOAGPDevice
  - IOPCIBridge
  - IOCardBusDevice

I/O Kit Generic PCI Driver

Josh de Cesare
Silicon Psychiatrist for Mac OS X

I/O Kit Drivers

- Passive and active matching
- Initialization
- Event processing
- User clients
- Access from applications
- Driver delivery

I/O Kit Drivers

- Passive matching—IOKitPersonalities
  - Matching properties
    - IOProviderClass
    - IONameMatch
    - IOPCIMatch
    - Others… (see IOKit/IOKitKeys.h)
  - IOProbeScore

I/O Kit Drivers

- Active Matching—driver::probe
  - Allows driver access to device's hardware
  - Hardware must be left in an "OK" state
  - Must free any used resources
  - Can bind multiple devices to one driver instance
  - Not normally needed

I/O Kit Drivers

- Initialization—driver::start
  - The provider—IOPCIDevice
    - Map Memory and I/O Space
      - getDeviceMemoryWithRegister
      - ioDeviceMemory
    - Configuration Cycles
      - configRead32, configWrite32, etc…
      - setMemoryEnable, setBusMasterEnable, etc…

I/O Kit Drivers

- Initialization—driver::start
  - Allocate IOWorkLoop and IOEventSources
    - IOInterruptEventSource
    - IOFilterInterruptEventSource
    - IOTimerEventSource
    - IOCommandGate
  - IOCommandPool
  - registerService

I/O Kit Drivers

- registerService
  - Makes the driver instance available for matching
  - Also enables user space matching and user clients
  - Can be used to signal that the driver is ready for use

I/O Kit Drivers

- Event Processing
  - IOWorkLoop
    - Serializes access to a driver's resources
    - Can Provides a thread for calling IOEventSource Actions
    - Can be paused as needed by the I/O stack
  - IOEventSource
    - Triggered by various sources
    - Executes their Action on a client thread or the IOWorkLoop thread depending on the type of IOEventSource

I/O Kit Drivers

- IOInterruptEventSource
  - A proxy for the device interrupt
  - The Action is a thread based interrupt handler
- IOFilterInterruptEventSource
  - Same features as IOInterruptEventSource, plus…
  - Allows a filter function at interrupt context

I/O Kit Drivers

- IOTimerEventSource
  - Executes an Action after a given time
  - Used for timeouts or signaling periodic events
- IOCommandGate
  - Executes a function on the current thread as if it were on the IOWorkLoop

I/O Kit Drivers

- IOCommandPool
  - Pre-allocate IOCommand subclasses for later use
    - Important for the drivers in the paging path
  - Can block a client until a command is free
  - Is synchronized with the IOWorkLoop

I/O Kit Drivers

- Property Access and User Clients
  - Get or set the driver's properties
    - Easy to use, connectionless, high overhead, low bandwidth
    - serializeProperties
    - setProperties
  - IOUserClient
    - Complicated, connection based, low overhead, high bandwidth
    - gIOUserClientClassKey

I/O Kit Drivers

- Access from Carbon or Cocoa Applications
  - IOKit.framework
  - Use matching functions to find the driver
  - Get or set the driver's properties
    - IORegistryEntryCreateCFProperties
    - IORegistryEntrySetCFProperties
  - IOUserClient
    - IOServiceOpen
  - Provide a library or CFPlugin as desired

Driver Delivery

- Driver Formats
  - .kext is the standard CFBundle form
    - Directory based, not a flat file
    - Not good for ROM use
  - .mkext is multi-kext
    - Flat file, compressed and checksummed
    - Only contains the Info.plist and driver's binary

Driver Delivery

- Creating a .mkext
  - Strip the debugging symbols from the driver

    strip -x mydriver.kext/Contents/MacOS/mydriver
  - mkextcache tool—compress and encode the driver

    mkextcache -o mydriver.mkext mydriver.kext

Driver Delivery

- FCode Drivers
  - Basically the same FCode as needed for Mac OS 9
  - Two new properties
    - "driver,AAPL,MacOSX,PowerPC"
      - Driver will be in the device tree
      - Use encode-file in the Fcode
      - **encode-file mydriver.mkext " driver,AAPL,MacOSX,PowerPC" property**
    - "driver-reg,AAPL,MacOSX,PowerPC"
      - Driver is the expansion ROM, only a reference in the device tree

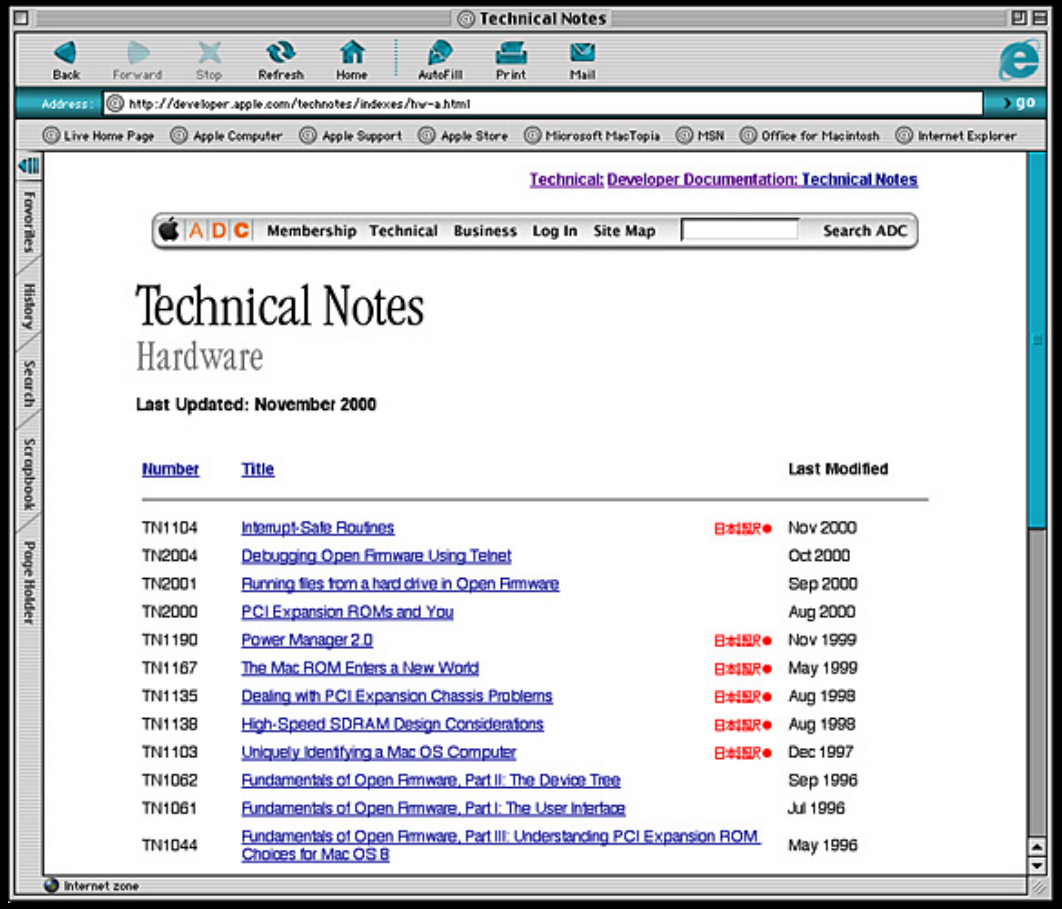
Driver Delivery

- Firmware Updates
  - Hardware access must be done by driver
  - Firmware Update Application is only a pretty front end
  - Firmware Image passed in chunks using set property interface
  - Driver writes chunks for FLASH and updates progress property
  - Application monitors progress and updates display

Where to Find Driver Examples?

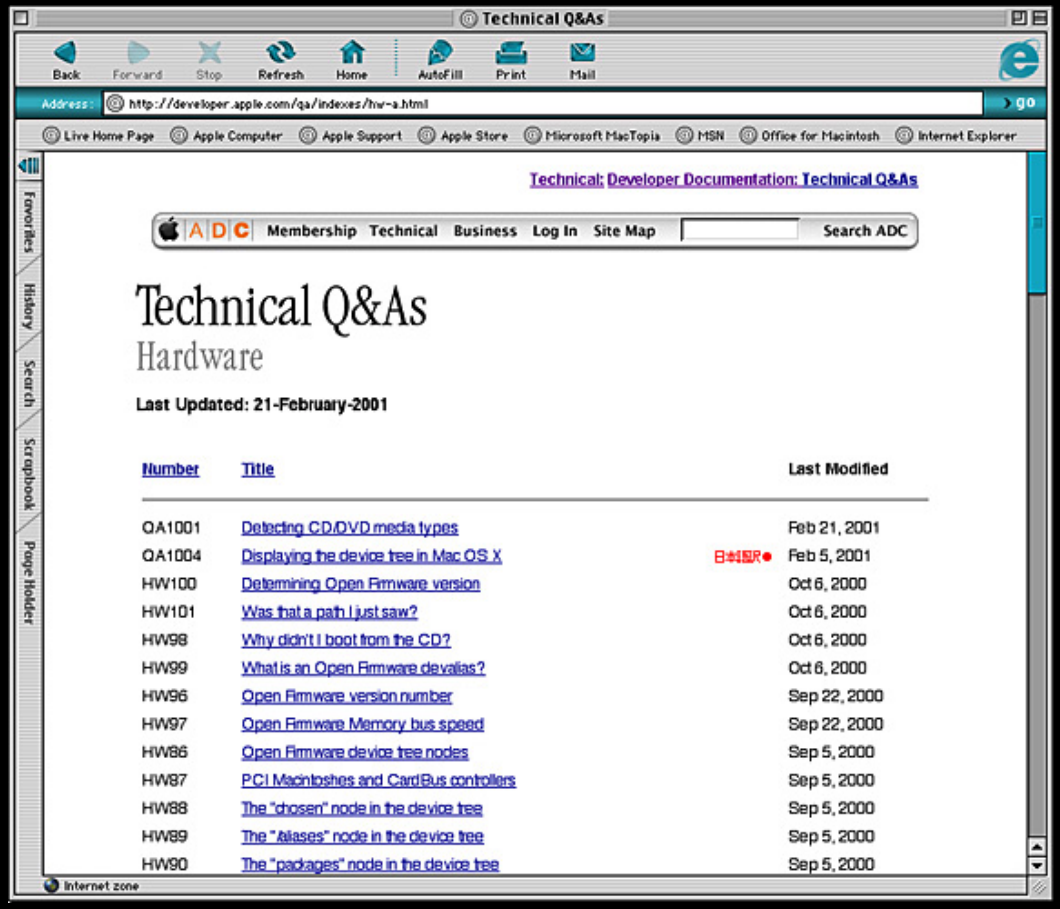
- Darwin
  - xnu
    - The kernel including I/O Kit source and headers
  - IOKitExamples/Kernel/IOKit/Generic/GenericPCI
    - A generic PCI driver
  - AppleBt8xx
    - A TV Card driver with application

Technical Notes



http://developer.apple.com/technotes/indexes/hw-a.html

Technical Q&As



http://developer.apple.com/qa/indexes/hw-a.html

Q&A

Mark Tozer-Vilchez
Desktop Technology Manager
tozer@apple.com

http://developer.apple.com/events/urls.html

WWDC

Worldwide Developers Conference 2001